Step 1:

Install RabbitMQ through docker

sudo docker run -d --hostname my-rabit --name ecomm-rabbit -p 15672:15672 -p 5672:5672 rabbitmq:3-management

15672 is used by RabbitMQ

5672 we will use in our program

Docker logs –f 3digitofimage

**Sender program**

using Newtonsoft.Json;

using RabbitMQ.Client;

using System;

using System.Text;

namespace ProdcuerDemo

{

class Program

{

static void Main(string[] args)

{

var factory = new ConnectionFactory() { HostName = "172.17.0.1" };

using(var connection = factory.CreateConnection())

using(var channel = connection.CreateModel()) {

channel.QueueDeclare(queue: "hello", durable: true, exclusive: false, autoDelete: false, arguments:

null);

string message = "Hello World!"; var body = Encoding.UTF8.GetBytes(message); channel.BasicPublish(exchange: "", routingKey: "hello",

basicProperties: null, body: body);

Console.WriteLine(" [x] Sent {0}", message);

}

Console.WriteLine(" Press [enter] to exit.");

Console.ReadLine();

}

}

}

**Consumer Program**

Program.cs

namespace Consumer

{

class Program

{

static void Main(string[] args)

{

var factory = new ConnectionFactory() { HostName = "172.17.0.1" }; using(var connection = factory.CreateConnection())

using(var channel =connection.CreateModel())

{

channel.QueueDeclare(queue: "hello", durable: true, exclusive: false, autoDelete: false, arguments: null);

Console.WriteLine(" [\*] Waiting for messages.");

var consumer = new EventingBasicConsumer(channel);

consumer.Received += (model, ea) => {

var body = ea.Body.ToArray();

var message = Encoding.UTF8.GetString(body);

Console.WriteLine(" [x] Received {0}", message);

};

channel.BasicConsume(queue: "hello", autoAck: true, consumer: consumer);

Console.WriteLine(" Press [enter] to exit.");

Console.ReadLine();

}

}

}

}

Here 172.12.1.0 is the IPAddress of my Ubuntu machine where docker & RabbitQueue is installed

**sudo ip addr show**

**2nd**

**Consumer**

using System;

using RabbitMQ.Client;

using RabbitMQ.Client.Events;

using System.Text;

using System.Threading;

class Worker

{

public static void Main()

{

var factory = new ConnectionFactory() { HostName = "localhost" };

using(var connection = factory.CreateConnection())

using(var channel = connection.CreateModel())

{

channel.QueueDeclare(queue: "task\_queue",

durable: true,

exclusive: false,

autoDelete: false,

arguments: null);

channel.BasicQos(prefetchSize: 0, prefetchCount: 1, global: false);

Console.WriteLine(" [\*] Waiting for messages.");

var consumer = new EventingBasicConsumer(channel);

consumer.Received += (sender, ea) =>

{

var body = ea.Body.ToArray();

var message = Encoding.UTF8.GetString(body);

Console.WriteLine(" [x] Received {0}", message);

int dots = message.Split('.').Length - 1;

Thread.Sleep(dots \* 1000);

Console.WriteLine(" [x] Done");

*// Note: it is possible to access the channel via*

*// ((EventingBasicConsumer)sender).Model here*

channel.BasicAck(deliveryTag: ea.DeliveryTag, multiple: false);

};

channel.BasicConsume(queue: "task\_queue",

autoAck: false,

consumer: consumer);

Console.WriteLine(" Press [enter] to exit.");

Console.ReadLine();

}

}

}

**Producer**

using System;

using RabbitMQ.Client;

using System.Text;

class NewTask

{

public static void Main(string[] args)

{

var factory = new ConnectionFactory() { HostName = "localhost" };

using(var connection = factory.CreateConnection())

using(var channel = connection.CreateModel())

{

channel.QueueDeclare(queue: "task\_queue",

durable: true,

exclusive: false,

autoDelete: false,

arguments: null);

var message = GetMessage(args);

var body = Encoding.UTF8.GetBytes(message);

var properties = channel.CreateBasicProperties();

properties.Persistent = true;

channel.BasicPublish(exchange: "",

routingKey: "task\_queue",

basicProperties: properties,

body: body);

Console.WriteLine(" [x] Sent {0}", message);

}

Console.WriteLine(" Press [enter] to exit.");

Console.ReadLine();

}

private static string GetMessage(string[] args)

{

return ((args.Length > 0) ? string.Join(" ", args) : "Hello World!");

}

}

[**https://www.rabbitmq.com/getstarted.html**](https://www.rabbitmq.com/getstarted.html)